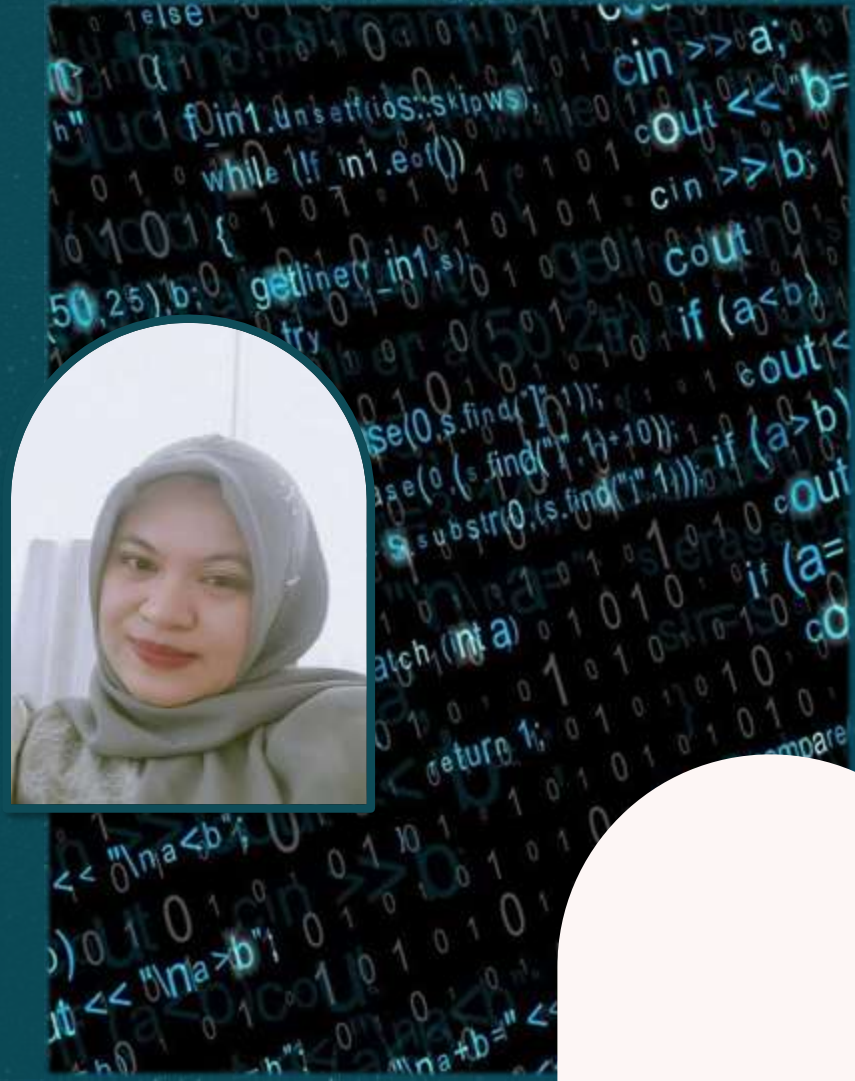
A SIMPLE CLASSIFICATION OF BREAST CANCER WISCONSIN USING RANDOM FOREST ALGORITHM

NURUL IZZA PUTRI

ABOUT ME

I am Nurul Izza Putri, Fresh graduate majoring in Information System from University of Jambi, Currently focusing on data sciences, data analyst and technological innovation. With strong skills in Microsoft office, Rstudio, google Colab and Tableau. Enthusiastic about combining creativity and analytical thinking to create innovative solutions in the field of information technology. I'm always lookinh for opportunities to grow by working on hand-on project and learning new skill.

With my academic backround, I've gained experience in data analyst, web development and design digital. I'm always enthusiastic about learning and embracing new challenges while continuosly expanding my exoertise in analytic and other area.



TOOL USED

BREAST CANCER

ABOUT DATASET

The dataset used in this project is the breast cancer Wisconsin (diagnostic) dataset from scikit-learn. It has 569 data sample with 30 numberical features representing breast tumor characteristics. This dataset aims to classify between benign (0) and malignant (1) tumors using random forest algortithm.

This dataset is widely used in testing machine learning models, especially in the health sector. This dataset is often used because of its relevance and good data structure.

RANDOM FOREST

Random Forest is an ensemble machine learning algorithm used for tasks like classification, regression and others. It works by creating multiple decision trees during training and combining their outputs to make more accurate and stable prediction.

in summary, random forest is a powerful, flexible algorithm suitable for both classification andd regression tasks, with strong performance and robustness against overfitting. However, it may not always be the best choice for real-time application due to its computational demands

Table of contents

IMPORT LIBRARY AND LOAD DATA

```
import pandas as pd
from sklearn import datasets

# Memuat dataset Wine dari scikit-learn dan mengonversinya menjadi DataFrame
cancer = datasets.load_breast_cancer()

X = cancer.data    # inputan untuk machine learning
y = cancer.target  # output yang dinginkan dari machine learning

# Mengonversi data fitur dan target menjadi DataFrame
df_X = pd.DataFrame(X, columns=cancer.feature_names)
df_y = pd.Series(y, name='target')

# Gabungkan fitur dan target dalam satu DataFrame
df = pd.concat([df_X, df_y], axis=1)

df.head(10)
```

IMPORT LIBRARY AND LOAD DATA

The data contains 569 data samples with 30 numerical features that describe the biological characteristics of the tumor (such as a radius, texture, area and smoothness). The targets are two classes, namely benign and malignant

#	Column	Non-Null Count	Dtype
0	mean radius	569 non-null	float64
1	mean texture	569 non-null	float64
2	mean perimeter	569 non-null	float64
3	mean area	569 non-null	float64
4	mean smoothness	569 non-null	float64
5	mean compactness	569 non-null	float64
6	mean concavity	569 non-null	float64
7	mean concave points	569 non-null	float64
8	mean symmetry	569 non-null	float64
9	mean fractal dimension	569 non-null	float64
10	radius error	569 non-null	float64
11	texture error	569 non-null	float64
12	perimeter error	569 non-null	float64
13	area error	569 non-null	float64
14	smoothness error	569 non-null	float64
15	compactness error	569 non-null	float64
16	concavity error	569 non-null	float64
17	concave points error	569 non-null	float64
18	symmetry error	569 non-null	float64
19	fractal dimension error	569 non-null	float64
20	worst radius	569 non-null	float64
21	worst texture	569 non-null	float64
22	worst perimeter	569 non-null	float64
23	worst area	569 non-null	float64
24	worst smoothness	569 non-null	float64
25	worst compactness	569 non-null	float64
26	worst concavity	569 non-null	float64
27	worst concave points	569 non-null	float64
28	worst symmetry	569 non-null	float64
29	worst fractal dimension	569 non-null	float64
30	target	569 non-null	int64

	mean radius	mean texture	mean perimeter	mean area	mean smoothness	mean compactness	mean concavity	mean concave points	mean symmetry	mean fractal dimension	...	worst texture	worst perimeter	worst area	worst smoothness	worst compactness	worst concavity	worst concave points	worst symmetry	worst fractal dimension	target
0	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.30010	0.14710	0.2419	0.07871	...	17.33	184.60	2019.0	0.1622	0.6656	0.7119	0.2654	0.4601	0.11890	0
1	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.08690	0.07017	0.1812	0.05667	...	23.41	158.80	1956.0	0.1238	0.1866	0.2416	0.1860	0.2750	0.08902	0
2	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.19740	0.12790	0.2069	0.05999	...	25.53	152.50	1709.0	0.1444	0.4245	0.4504	0.2430	0.3613	0.08758	0
3	11.42	20.38	77.58	386.1	0.14250	0.28390	0.24140	0.10520	0.2597	0.09744	...	26.50	98.87	567.7	0.2098	0.8663	0.6869	0.2575	0.6638	0.17300	0
4	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.19800	0.10430	0.1809	0.05883	...	16.67	152.20	1575.0	0.1374	0.2050	0.4000	0.1625	0.2364	0.07678	0
5	12.45	15.70	82.57	477.1	0.12780	0.17000	0.15780	0.08089	0.2087	0.07613	...	23.75	103.40	741.6	0.1791	0.5249	0.5355	0.1741	0.3985	0.12440	0
6	18.25	19.98	119.60	1040.0	0.09463	0.10900	0.11270	0.07400	0.1794	0.05742	...	27.66	153.20	1606.0	0.1442	0.2576	0.3784	0.1932	0.3063	0.08368	0
7	13.71	20.83	90.20	577.9	0.11890	0.16450	0.09366	0.05985	0.2196	0.07451	...	28.14	110.60	897.0	0.1654	0.3682	0.2678	0.1556	0.3196	0.11510	0
8	13.00	21.82	87.50	519.8	0.12730	0.19320	0.18590	0.09353	0.2350	0.07389	...	30.73	106.20	739.3	0.1703	0.5401	0.5390	0.2060	0.4378	0.10720	0
9	12.46	24.04	83.97	475.9	0.11860	0.23960	0.22730	0.08543	0.2030	0.08243	...	40.68	97.65	711.4	0.1853	1.0580	1.1050	0.2210	0.4366	0.20750	0

EXPLORATORY DATA ANALYSIS (EDA)

```
#mengidentifikasi semua number yang berbeda
df['target'].unique()
```

```
array([0, 1])
```

```
[ ] #menampilkan statik dari data
    df.describe()
```

	mean radius	mean texture	mean perimeter	mean area	mean smoothness	mean compactness	mean concavity	mean concave points	mean symmetry	mean fractal dimension	...	worst texture	worst perimeter	worst area	worst smoothness	worst compactness	worst concavity	co p
count	569.000000	569.000000	569.000000	569.000000	569.000000	569.000000	569.000000	569.000000	569.000000	569.000000	...	569.000000	569.000000	569.000000	569.000000	569.000000	569.000000	569.0
mean	14.127292	19.289649	91.969033	654.889104	0.096360	0.104341	0.088799	0.048919	0.181162	0.062798	...	25.677223	107.261213	880.583128	0.132369	0.254265	0.272188	0.1
std	3.524049	4.301036	24.298981	351.914129	0.014064	0.052813	0.079720	0.038803	0.027414	0.007060	...	6.146258	33.602542	569.356993	0.022832	0.157336	0.208624	0.0
min	6.981000	9.710000	43.790000	143.500000	0.052630	0.019380	0.000000	0.000000	0.106000	0.049960	...	12.020000	50.410000	185.200000	0.071170	0.027290	0.000000	0.0
25%	11.700000	16.170000	75.170000	420.300000	0.086370	0.064920	0.029560	0.020310	0.161900	0.057700	...	21.080000	84.110000	515.300000	0.116600	0.147200	0.114500	0.0
50%	13.370000	18.840000	86.240000	551.100000	0.095870	0.092630	0.061540	0.033500	0.179200	0.061540	...	25.410000	97.660000	686.500000	0.131300	0.211900	0.226700	0.0
75%	15.780000	21.800000	104.100000	782.700000	0.105300	0.130400	0.130700	0.074000	0.195700	0.066120	...	29.720000	125.400000	1084.000000	0.146000	0.339100	0.382900	0.1
max	28.110000	39.280000	188.500000	2501.000000	0.163400	0.345400	0.426800	0.201200	0.304000	0.097440	...	49.540000	251.200000	4254.000000	0.222600	1.058000	1.252000	0.2

8 rows × 31 columns

DATA MODELLING

```python
from sklearn.model_selection import train_test_split

# Membagi data menjadi train dan test
X_train, X_test, y_train, y_test = train_test_split(df_X, df_y, test_size=0.2, random_state=42)
```

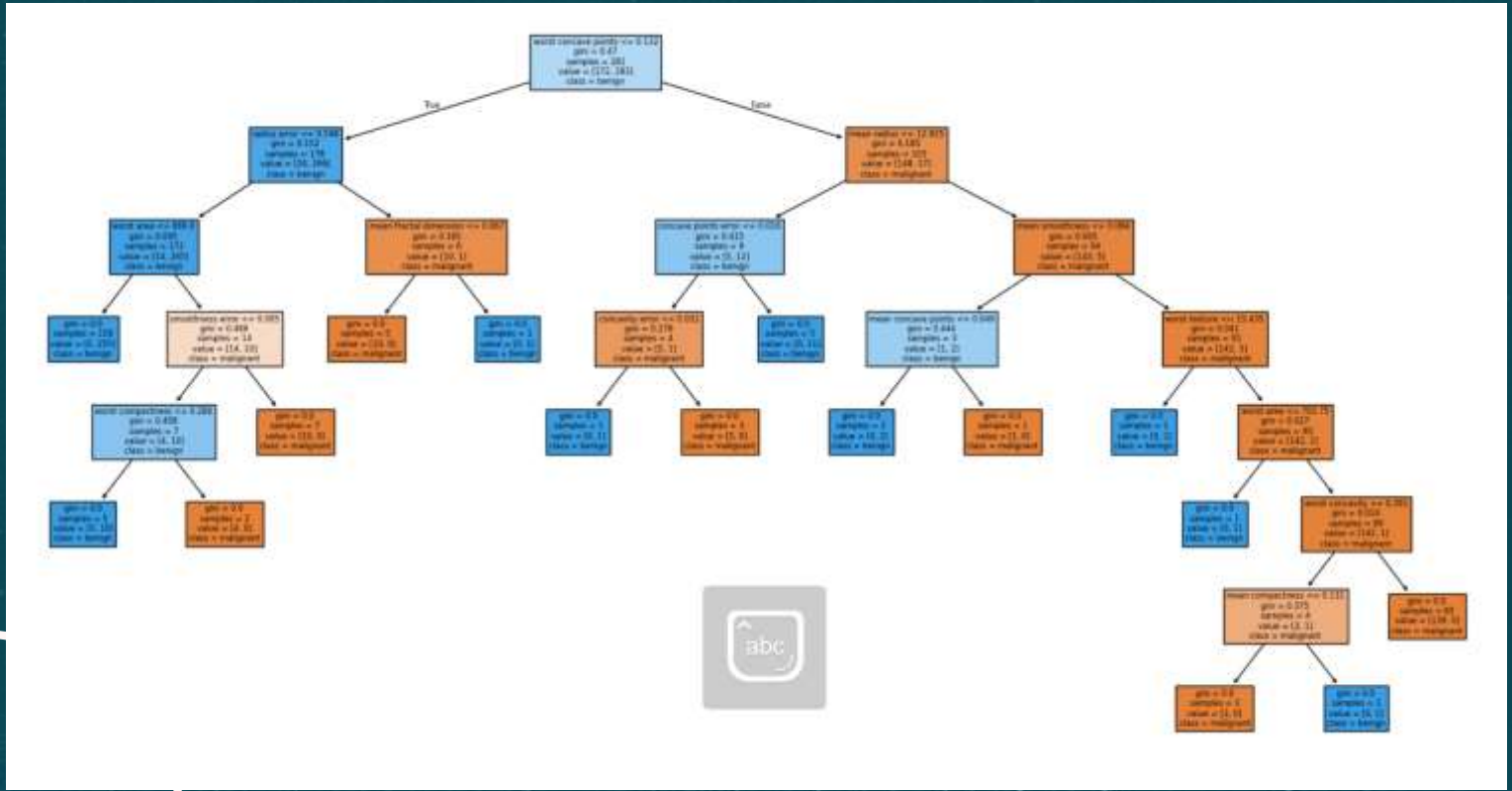
```python
from sklearn.ensemble import RandomForestClassifier

# Membuat dan melatih model Decision Tree
model = RandomForestClassifier(n_estimators=100, random_state=42)
model.fit(X_train, y_train)
```

```
RandomForestClassifier
RandomForestClassifier(random_state=42)
```

```python
from sklearn.metrics import accuracy_score, classification_report

# melakukan prediksi dan evaluasi model
y_pred = model.predict(X_test)

accuracy = accuracy_score(y_test, y_pred)

print("classification Report:")
print(classification_report(y_test, y_pred))
print(f"Accuracy: {accuracy * 100:.2f}%")
```

```
classification Report:
              precision    recall  f1-score   support

           0       0.98      0.93      0.95        43
           1       0.96      0.99      0.97        71

    accuracy                           0.96       114
   macro avg       0.97      0.96      0.96       114
weighted avg       0.97      0.96      0.96       114

Accuracy: 96.49%
```

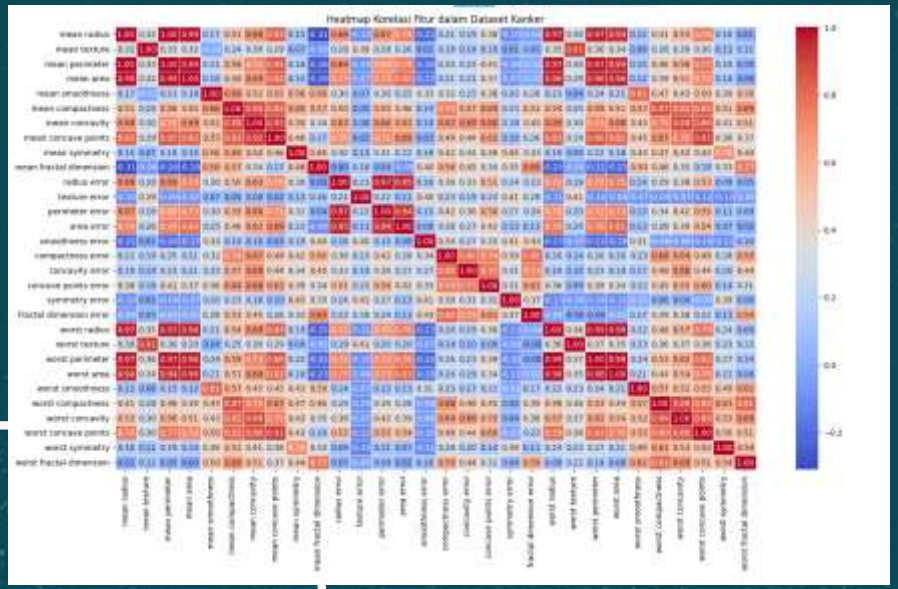
DECISION TREE RESULT



DATA VISUALIZATION

Correlation Matrix

```
# Visualisasi Heatmap korelasi antar fitur
plt.figure(figsize=(18, 10))
correlation_matrix = df.drop('target', axis=1).corr() #menghitung korelasi antar fitur
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt='.2f', linewidths=0.5)
plt.title('Heatmap Korelasi Fitur dalam Dataset Kanker')
plt.show()
```

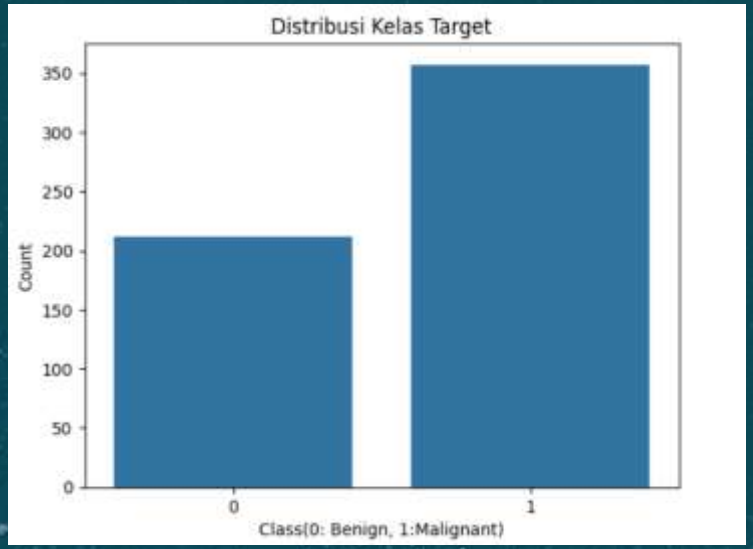


The correlation matrix show the correlation between features between the existing in a dataset. High or low correlation between features can provide insight into the relationship between existing features. By visualizing the correlation will be used to build a machine learning model.

DATA VISUALIZATION

Distribution of Target Classes

```python
import matplotlib.pyplot as plt
import seaborn as sns

# visualisasi distribusi kelas target menggunakan seaborn
sns.countplot(x='target', data=df)
plt.title('Distribusi Kelas Target')
plt.xlabel('Class(0: Benign, 1:Malignant)')
plt.ylabel('Count')
plt.show()
```

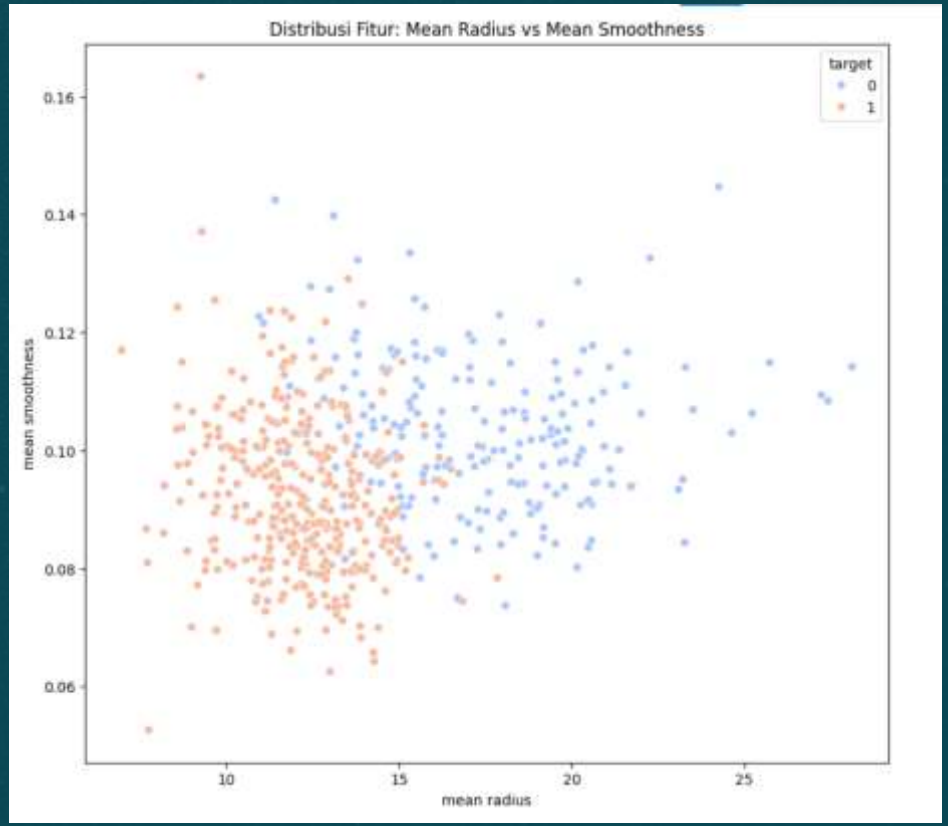


Distribution of Target Classes: Displays the amount of data for each class (Benign and malignant) using a bar chart (countplot).

DATA VISUALIZATION

Scatterplot

```
# Visualisasi Distribusi dari beberapa fitur

plt.figure(figsize=(10,10))
sns.scatterplot(x='mean radius', y='mean smoothness', hue='target', data=df, palette='coolwarm')
plt.title('Distribusi Fitur: Mean Radius vs Mean Smoothness')
plt.show()
```



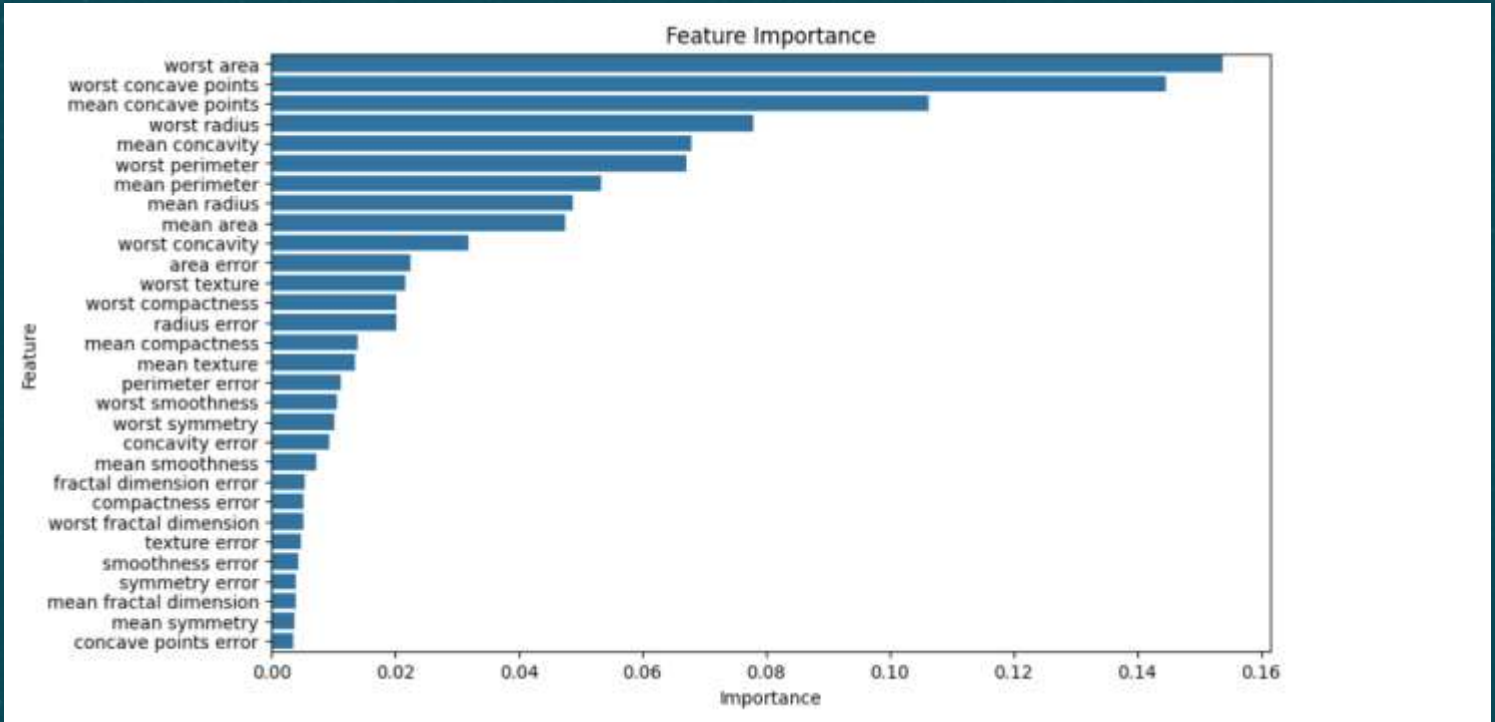
DATA VISUALIZATION

Feature Importance

```
# Visualisasi pentingnya fitur dari model Random Forest
importances = model.feature_importances_

# Access feature names from the breast cancer dataset's feature_name attribute
feature_names = cancer.feature_names

feature_importances_df = pd.DataFrame({'Feature' : feature_names, 'Importance': importances})
feature_importances_df = feature_importances_df.sort_values(by='Importance', ascending=False)

plt.figure(figsize=(10,6))
sns.barplot(x='Importance', y='Feature', data=feature_importances_df)
plt.title('Feature Importance')
plt.xlabel('Importance')
plt.ylabel('Feature')
plt.show()
```

DATA VISUALIZATION

Feature Importance



Thank You

Nrl1511_

nurul-izza-putri-842395349

Github.com/nrlizzaputri